MISSING VALUES IN NONLINEAR FACTOR ANALYSIS

Tapani Raiko and Harri Valpola

Helsinki University of Technology, Neural Networks Research Centre
P.O.Box 5400, FIN-02015 HUT, Espoo, Finland
E-mail: `Tapani.Raiko@hut.fi, Harri.Valpola@hut.fi`
URL: `http://www.cis.hut.fi/`

ABSTRACT

The properties of the nonlinear factor analysis (NFA) model are studied by measuring how well it reconstructs missing values in observations. The NFA model uses a multi-layer perceptron (MLP) network for approximating the nonlinear mapping from factors to observations. The NFA model is compared with linear factor analysis (FA) and with the self-organising map (SOM). The number of parameters in the NFA model is closer to FA than the SOM, but unlike FA, NFA is able to model nonlinear manifolds. Based on experiments with real world speech data and Boston housing data, we conclude that the performance of the NFA model is closer to FA.

1. INTRODUCTION

Generative models handle missing values in an easy and natural way. Whenever a model is found, reconstructions of the missing values are also obtained. Generative models are not the only way to handle the missing data [8, 10, 11], but we only cover them in this paper. Unsupervised learning can be used for supervised learning by considering the outputs of the test data as missing values (Figure 1). This combines feature extraction and supervised learning.

The ability to reconstruct missing values measures the quality of a model and its ability to generalise. Reconstructions are used in this paper to demonstrate the properties of nonlinear factor analysis (NFA) [6] by comparing it to linear factor analysis (FA) and to the self-organising map (SOM) [5].

FA is like principal component analysis (PCA) with modelled noise. It is a basic tool that works well when nonlinear effects are not important. Large dimensionality of data is not a problem. The SOM captures nonlinearities and clusters, but has difficulties with data of high intrinsic dimensionality and generalisation. NFA has properties of both of them; It can handle both large dimensionality and nonlinear effects.

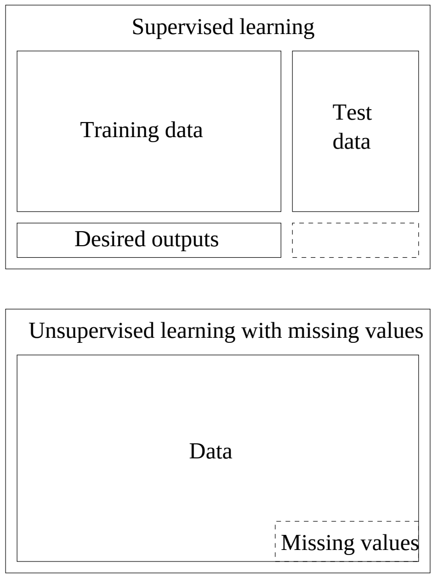

Figure 1: Unsupervised learning can be used for supervised learning by considering the outputs of the test data as missing values.

The three methods are described shortly in the next two sections. The fourth section describes the experiments and the results. They are discussed in section five.

2. SUPERVISED SELF-ORGANISING MAP

A self-organising map, which is used to reconstruct missing values, is called the Supervised SOM [5]. Model vectors lie in the same space as the data. Here we use a variant in which each data vector is matched to model vectors ignoring the values that are missing. In the learning phase, the winning model vector and its neighbours in the map are moved slightly towards the data vector ignoring the dimensions of the missing values again. Finally the reconstructions of the missing values are combinations of the values from the model vectors, which are weighted according to Gaussian kernels assigned to them:

$$\hat{\mathbf{x}}(t) = \frac{\sum_i \mathbf{g}(\mathbf{x}(t), \mathbf{m}_i, \sigma^2)\mathbf{m}_i}{\sum_i \mathbf{g}(\mathbf{x}(t), \mathbf{m}_i, \sigma^2)} \tag{1}$$

$$\mathbf{g}(\mathbf{x}, \mathbf{m}, \sigma^2) \propto \exp -\frac{d(\mathbf{x}, \mathbf{m})^2}{2\sigma^2} \tag{2}$$

$$d(\mathbf{x}, \mathbf{m}) = \sqrt{\sum_{k \text{ observed in } \mathbf{x}} (x_k - m_k)^2} \tag{3}$$

$\mathbf{x}(t)$ is a data vector and $\hat{\mathbf{x}}(t)$ its reconstruction. $\mathbf{m}_i$ is a model vector. Parameter σ is called the width of the softening kernels. When σ approaches zero, the reconstruction approaches the single winning model vector. The data set might contain missing values in each of the vectors, but the model vectors contain no missing values.

The reconstructions are restricted to convex combinations of the model vectors. This might be significant, when the true dimensionality of the data is large and most of the data is on the border: When a three dimensional ball loses 5 percent of its radius, it loses about 14 percent of its mass, but a 50 dimensional ball loses more than 92 percent of its mass.

3. LINEAR AND NONLINEAR FACTOR ANALYSIS

3.1. Model structure

According to the general FA model the data has been generated by factors $\mathbf{s}$ through mapping $\mathbf{f}$:

$$\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t), \boldsymbol{\theta}) + \mathbf{e}(t) \;, \tag{4}$$

where $\mathbf{x}$ is a data vector, $\mathbf{s}$ is a factor vector, $\boldsymbol{\theta}$ is a parameter vector and $\mathbf{e}$ is a noise vector. The linear mapping $\mathbf{f}$ used in FA is

$$\mathbf{f}(\mathbf{s}, \boldsymbol{\theta}) = \mathbf{A}\mathbf{s} + \mathbf{b} \;. \tag{5}$$

The model is similar to principal component analysis except that FA includes the noise term and the factors have a Gaussian distribution. In NFA, the function $\mathbf{f}$ is allowed to be nonlinear. We use the method proposed in [6], where the MLP network

$$\mathbf{f}(\mathbf{s}, \boldsymbol{\theta}) = \mathbf{A}_2 \tanh(\mathbf{A}_1\mathbf{s} + \mathbf{b}_1) + \mathbf{b}_2 \tag{6}$$

is used to model the nonlinearity. The parameter vector $\boldsymbol{\theta}$ contains both $\mathbf{A}$ and $\mathbf{b}$. The factors and the noise are assumed to be independent and Gaussian.

In NFA the data is modelled by a high dimensional manifold created by function $\mathbf{f}$ from a prior Gaussian distribution. It can be compared to the self-organising map (SOM) [5], but the number of parameters scale more like in FA. The SOM scales exponentially as function of the dimensionality of the underlying data manifold. A small number of parameters keeps the modelled manifold smooth. We find the parameter vector $\boldsymbol{\theta}$ using ensemble learning.

3.2. Ensemble learning

In general there are infinitely many possible explanations of different complexity for the observed data. Choosing too complex a model results in overfitting, where the model tries to make up meaningless explanations for the noise in addition to the true factors. Choosing too simple a model results in underfitting, leaving hidden some of the true factors that have generated the data.

The solution to the problem is that no single model should actually be chosen. Instead, all the possible explanations should be taken into account and weighted according to their posterior probabilities. This approach, known as Bayesian learning, optimally solves the tradeoff between under- and overfitting.

In practice, exact treatment of the posterior pdfs of the models is impossible. Therefore, some suitable approximation method must be used. Ensemble learning [4, 1, 7, 9], which is one type of variational learning, is a method for parametric approximation of posterior pdfs. The basic idea in ensemble learning is to minimise the misfit between the posterior pdf and its parametric approximation.

Let $P(\theta|\mathbf{x})$ denote the exact posterior pdf and $Q(\theta)$ its parametric approximation. The misfit is measured with the Kullback-Leibler (KL) divergence C_{KL} between P and Q, defined by the cost function

$$\begin{aligned} C_{\mathrm{KL}} &= E_Q\left\{\log\frac{Q}{P}\right\} \qquad (7) \\ &= \int Q(\mathbf{s},\theta)\log\frac{Q(\mathbf{s},\theta)}{P(\mathbf{s},\theta|\mathbf{x})}d\theta d\mathbf{s} \\ &= \int Q(\mathbf{s},\theta)\log\frac{Q(\mathbf{s},\theta)}{P(\mathbf{s},\theta,\mathbf{x})}d\theta d\mathbf{s} + \log P(\mathbf{x}). \end{aligned}$$

Because the KL divergence involves an expectation over a distribution, it is sensitive to probability

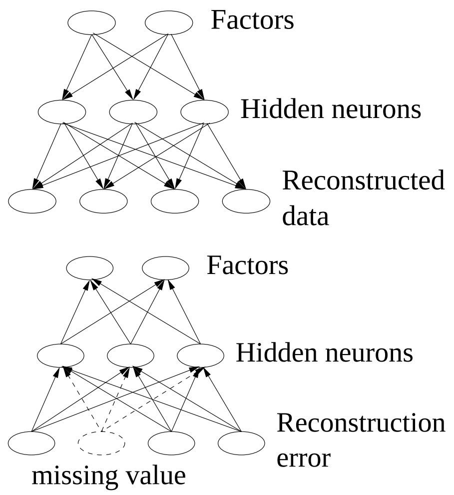

Figure 2: Upper figure: MLP net as a generative model, Lower figure: Error propagates only from observed values to factors.

mass rather than to probability density. The term $\log P(\mathbf{x})$ does not depend on the parameters or the factors and can be neglected.

The learning resembles the expectation maximisation (EM) algorithm. The factors are adjusted while keeping the mapping constant and the mapping is adjusted while keeping the factors constant always minimising the cost function. All the parameters and factors are modelled with Gaussian distributions rather than point estimates.

A more detailed account of the unsupervised ensemble learning method used for nonlinear factor analysis and discussion of potentially appearing problems can be found in [6].

3.3. Reconstruction

The approximation $Q(\mathbf{s}, \boldsymbol{\theta})$ is assumed to be a Gaussian density with a diagonal covariance matrix. This simplifies the cost function given by (7) into expectations of many simple terms. Some of them relate to the noise vectors $\mathbf{e}(t)$ which are also the reconstruction errors as seen in formula (4). In case of a missing value this term is not included in the cost function and the factors are thus estimated based only on the available observations. The reconstruction of data is obtained by the mapping $\mathbf{f}$ from the estimated factors (Figure 2).

There is an analogy with the SOM. Factor vector $\mathbf{s}$ corresponds to the winning map unit in the SOM and $\mathbf{f}(\mathbf{s})$ corresponds to the model vector of the winner.

3.4. Learning procedure

First, linear PCA (principal component analysis) is applied to find sensible initial values for the posterior means of the factors. PCA has been modified to accept missing values by calculating the covariances from only those pairs of data values where both values are observed (as opposed to missing). The posterior variances of the factors are initialised to small values.

The factors were fixed at the values given by linear PCA for the first 50 sweeps through the entire data set. This allows the network to find a meaningful mapping from factors to the observations, thereby justifying using the factors for the representation. For the same reason, the parameters controlling the distributions of the factors, weights, noise and the hyperparameters are not adapted during the first 100 sweeps. They are adapted only after the network has found sensible values for the variables whose distributions these parameters control. This setting is important for the method because the network can effectively prune away unused parts, which would lead to a local minimum from which the network would never recover.

4. EXPERIMENTS

4.1. Experiment settings

The experiment setting is to reconstruct the missing values and the mean square error of the reconstructions are used for the comparison. The two data sets that are used are speech data and Boston housing data. Ignorability of the data collection mechanism [3] is assumed in this paper. The collection mechanism is nonignorable, for instance, when out-of-scale measurements are marked as missing.

The first data set consists of real-world Finnish speech spectrograms spoken by several individuals. Short term spectra are windowed to 30 dimensions with a standard preprocessing procedure for speech recognition. It is clear that a dynamic model [12] would give better reconstructions, but in this case the temporal information is left out to ease the comparison of the models. Half of the about 5000 samples are used as test data with some missing values. Missing values are set in four different ways to measure different properties of the algorithms (Figure 3):

1. 38 percent of the values are set to miss randomly in 4 times 4 patches. (Figure 4) This is the main setting, since it is most realistic.

2. 10 percent of the values are set to miss randomly independent of any neighbours. This is

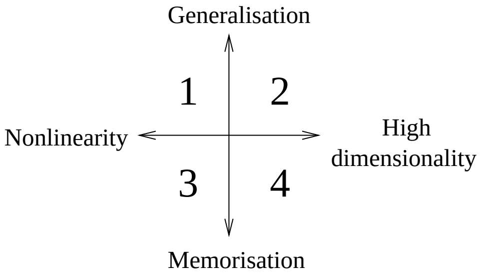

Figure 3: Four different experiment settings with the speech data try to measure different properties of the algorithms.

an easier setting, since simple smoothing using nearby values would give fine reconstructions.

3. Training and testing sets are randomly permuted before setting missing values in 4 times 4 patches as in setting 1. The training set contains vectors more similar to the test set now.

4. Training and testing sets are permuted and 10 percent of the values are set to miss independently of any neighbours.

The second data set is Boston housing data, which is publicly available at [2]. It concerns housing values in suburbs of Boston. Data set contains 506 vectors of 13 dimensions excluding one binary attribute. Four of the 13 values were common to each town, which consist of 1 to 50 suburbs. 70% of the data vectors are used as training data and the rest as testing data, which has 10% of its values missing randomly.

4.2. Implementation

Self-organising map was learned using the SOM Toolbox[1] for Matlab. The number of model vectors and the width of the softening kernels associated with them were left as parameters and the selection of values is not studied herein. Instead, the values with the best results were used. Number of iterations was the default value in the SOM toolbox and additional 50 iterations through all the data were run with neighbourhood set off.

The NFA code for Matlab was modified to support missing values. The code needed to reproduce the experiments is publicly available[2]. The NFA method was tried with fixed number of hidden neurons using several different number of factors only, because the algorithm requires about ten thousand batch iterations, where one batch iteration means going through all the observations once.

[1] http://www.cis.hut.fi/projects/somtoolbox/

[2] http://www.cis.hut.fi/projects/ica/bayes/

4.3. Performance with the speech data

The performance of different methods with speech data can be seen in Figure 5. NFA was tested with 30 hidden neurons and 2 to 15 factors and FA with number of factors varying from 1 to 30. The implemented NFA algorithm suffered from instability when the number of factors was greater than 15. The NFA model performed always better than FA with same number of factors.

The mean square reconstruction errors are collected here in the same order as in Figure 3:

NFA		FA		SOM	
1.76	0.57	1.88	0.57	1.73	0.83
1.73	0.57	1.85	0.58	1.52	0.85

The first setting proved to be the hardest as expected, since new words require generalisation and missing values in patches makes nonlinear effects more important. With optimal parameter values the SOM gave marginally better reconstructions than NFA. FA performed the worst.

The second setting was easier: NFA and FA performed equally well, but the SOM could not achieve same accuracy. A large number of model vectors did not help the SOM to get enough representing power.

The third setting had permuted data sets, which makes generalisation less important. This helped the SOM a lot and it gave clearly the best results. NFA and FA benefitted only marginally and were left behind.

Results of the fourth setting did not differ from the second setting. The change in the missing value pattern from the first and third settings seems to have the dominant effect.

The reconstruction errors of the observed values (lower curves in Figure 5) are of interest, too. The SOM can not represent the data as accurately with the map unit activities as the factor analysis models can with the factor values. The FA model with 30 factors could have represented the data perfectly, but the modelled noise accounted for some variation.

The best number of model vectors in the SOM was 1600 in the first case, but at least 2400 in the other cases. This has caused a change in the reconstruction error of the observed values. The optimal width of the softening kernel was also somewhat larger in the first experiment. The number of map units is normally not as large as half of the number of data vectors.

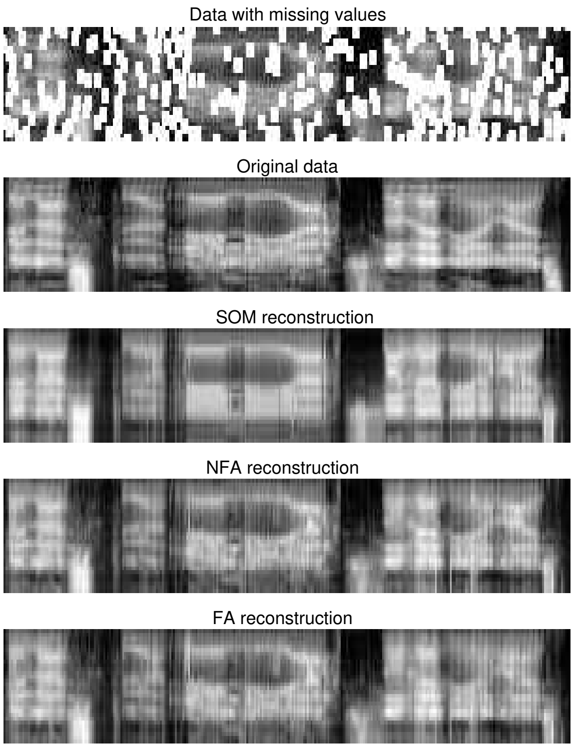

Figure 4: Speech data reconstruction example with best parameters of each algorithm.

4.4. Performance with the Boston data

The experiments with Boston data were run four times with random division of data into training and test sets. Mean square reconstruction errors were scaled such that the SOM errors were 100 in each case. The NFA errors were 118 with standard deviation of 17 and the FA errors were 151 with standard deviation of 22.

This data set is clustered, because of the town structure, and the dimensionality of the data manifold is not too large for the SOM to handle. Therefore it is not very surprising that it made better reconstructions than NFA. Nonlinearities were crucial in the data set, since FA was inferior to the nonlinear methods. NFA was run with 20 hidden neurons and from 1 to 9 factors. Best number of them varied from 5 to 9. The best number model vectors in the SOM varied from 500 to 1800, which is far greater than the number of data points.

5. DISCUSSION

Figures 6 and 7 show some projections of the data and the models. The SOM manifold looks quite curly. NFA manifold with two factors is also rolled up, but in a smoother way. Important differences between high dimensional NFA manifolds are not visible in these projections.

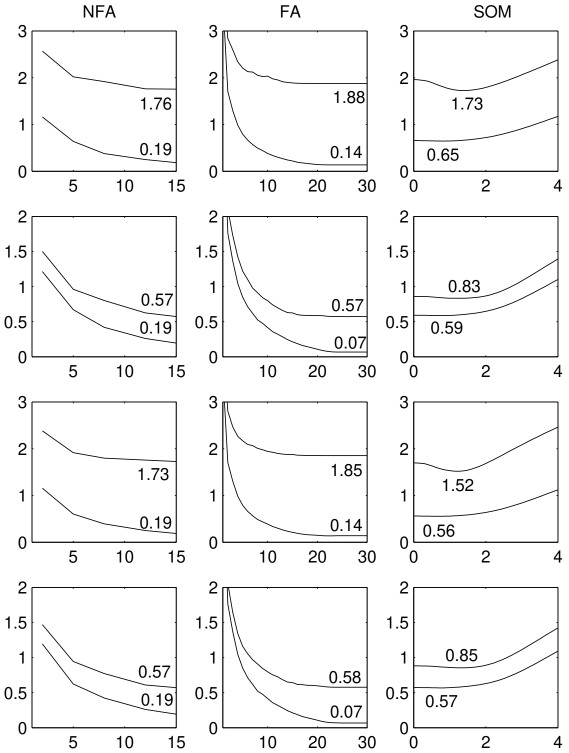

Figure 5: Mean square reconstruction errors of speech data as a function of number of factors or softening width. The rows correspond to the experiment settings reading from up to down. The upper curve in each plot is the test result for missing values and the lower curve is the reconstruction of observed values.

As expected, the SOM performs the best for clustered data and NFA performed better than FA. Results show that NFA is closer to FA than to the SOM. The greatest problem of the SOM is that the number of parameters scales exponentially with the number of intrinsic dimensions of the data manifold, which leads to bad generalisation. The NFA model does not work well, if the data is clustered. It is hard to find the function that shapes a Gaussian continuum to clusters, but when the data forms a continuum, too, the NFA model is more appropriate.

The NFA algorithm searches only for local optima, so multiple runs with different initialisations would have better chances globally. The stability should be guaranteed and some speedups could be made for the algorithm to be a ready-to-use tool.

Even though the learning algorithm requires a lot processing, the NFA model, when it has learned,

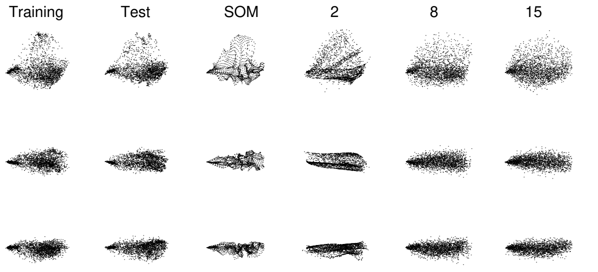

Figure 6: The speech data sets, the SOM model vectors and some points generated by NFA with varying number of factors are projected on three planes. In the uppermost row the plane is the 1st and 2nd principal components and in the middle the 1st and 3rd components and in lowermost row the 1st and 4th component. NFA models were used by generating random factor vectors from their prior distribution. The dots shown are the corresponding expected values of the data vectors.

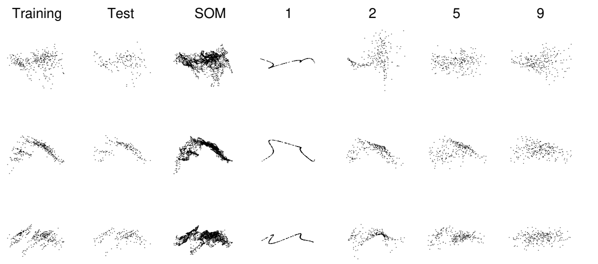

Figure 7: Boston housing data, projections as in Figure 6.

can be applied in real time. The computational complexity of the algorithm scales to the product of number of factors and hidden neurons per batch iteration. By contrast, the number of map units in a SOM scales exponentially as a function of the dimensionality of the map. NFA is best suited for fairly strongly nonlinear problems with an intrinsic dimension of the order of ten.

When compared to the SOM, factor analysis models are simpler in terms of the number of parameters. While even the largest NFA model had about 3000 parameters, the SOM had about 50000 parameters. This can be seen in Figure 6 from the fact that NFA did not capture all the finest details in the data set. Experimental results support the statement that small number of parameters enhances the ability to generalise.

6. CONCLUSIONS

The ability to reconstruct missing values measures the quality of a model: its ability to generalise, memorise and represent. The nonlinear factor analysis model turned out to perform well in generalisation and representation while its ability to memorise is limited due to the small number of parameters in the model. NFA performed better than FA in all the experiments, but with large number of factors, the current NFA algorithm becomes computationally expensive. We conclude that nonlinear factor analysis (NFA) is best suited for fairly strongly nonlinear problems with an intrinsic dimension of the order of ten. [3]

7. REFERENCES

[1] D. Barber and C. M. Bishop. Ensemble learning in Bayesian neural networks. In M. Jordan, M. Kearns, and S. Solla, editors, *Neural Networks and Machine Learning*, pages 215–237. Springer, Berlin, 1998.

[2] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[3] A. Gelman, J. Garlin, H. Stern, and D. Rubin. *Bayesian Data Analysis.* Chapman & Hall/CRC, 1995.

[4] G.E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proceedings of the COLT'93*, pages 5–13, Santa Cruz, California, USA, July 26–28, 1993.

[5] T. Kohonen. *Self-Organizing Maps.* Springer, 3rd, extended edition, 2001.

[6] H. Lappalainen and A. Honkela. Bayesian nonlinear independent component analysis by multilayer perceptrons. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 93–121. Springer, Berlin, 2000.

[7] H. Lappalainen and J. Miskin. Ensemble learning. In M. Girolami, editor, *Advances in Independent Component Analysis*, pages 75–92. Springer, Berlin, 2000.

[8] R.J.A. Little and D.B.Rubin. *Statistical Analysis With Missing Data.* J. Wiley & Sons, 1987.

[9] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. The MIT Press, Cambridge, Massachusetts, 1999.

[10] D.B. Rubin. *Multiple Imputation for Nonresponse in Surveys.* J. Wiley & Sons, 1987.

[11] J.L. Schafer. *Analysis of Incomplete Multivariate Data.* Chapman & Hall/CRC, 1997.

[12] H. Valpola. Unsupervised learning of nonlinear dynamic state-space models. Technical Report A59, Helsinki University of Technology, Espoo, Finland, 2000.

[3] This work has been funded by the EU project BLISS.